PROJECT REPORT

## NHPC CONTRACT SUBMISSION PORTAL

Submitted in fulfilment of the requirements for the five week training for the award of the degree of

## BACHELORS OF TECHNOLOGY INFORMATION TECHNOLOGY

from

## BHAGWAN PARSHURAM INSTITUTE OF TECHNOLOGY

For the duration

### 11th June 2025 to 16th July 2025

at

### NHPC Limited Corporate Office, Santosh Nagar

**Sector 33, Faridabad, Haryana 121003**

Submitted to Submitted by

## TAPAN BANERJEE SHAKSHAM

**DECLARATION**

I, **SHAKSHAM,** hereby declare that the presented report of training project titled **“NHPC CONTRACT SUBMISSION PORTAL”** is uniquely prepared by me after the completion of five weeks of industrial training at **IT&C Department, Corporate Office, NHPC Limited,** under the guidance and mentorship of **Mr. TAPAN**

### BANERJEE, Group Senior Manager (IT).

I also declare that the project work presented in this report has not been previously submitted by any others for any academic qualification or certificate and report has nothing in confidential in respect of the company of training. I take responsibility for all the legal and ethical requirements regarding this report.

### Shaksham Date: 16/07/2025

**ACKNOWLEDGEMENT**

The internship opportunity I had with **NHPC Limited** was a great opportunity for learning and professional development. Therefore, I would like to express my gratitude for this opportunity. I am also grateful for having a chance to meet so many wonderful people and professionals who led me throughout this internship period.

First, I would like to express my gratitude to my mentor at Corporate Office NHPC Limited, **Mr. TAPAN BANERJEE, GSM(IT),** who communicated daily tasks efficiently making our weekly milestones easy to reach. He helped me through the internship period creating the best possible environment to work efficiently.

I perceive this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on the improvement, to attain desired career objectives. Hope to continue cooperation with all of you in the future.

I am also grateful to **Dr Rajesh Kr Ahuja (Training & Placement officer)** for allowing me to perform the Industrial Training successfully. Expressing my heartiest gratitude to **Dr. Munish Vashisth (Chairperson of Electronics Department)** and for their constant encouragement and assistance during the project period.

**Shaksham**

**Date: 16/07/2025**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **TITLE** | **Page No.** |
| 1. | Chapter 01 Company Profile   * 1. Introduction to NHPC Limited   2. Objectives   3. Profile   4. Awards and Honours   5. IT&C Department, Corporate Office | 6-12 |
| 2. | Chapter 02 Introduction to Project   * 1. Overview   2. Working   3. Uses and Purpose | 13-15 |
| 3. | Chapter 03 Requirement Analysis   * 1. Functional Requirements   2. Non-Functional Requirements   3. System requirements | 16-19 |
| 4. | Chapter 04 Design Tools   * 1. High- Level Design   2. Data flow Design | 20-26 |
| 5. | Chapter 05 Module Implementation   * 1. Frontend UI and Interaction Module   2. Contract Submission Module   3. Contract Extension Module   4. Dashboard and Data Display Module   5. Backend API and Logic Module | 27-31 |
| 6. | Chapter 06 Database and Data Directory   * 1. Database schema   2. Entity-Relationship Discription | 32-38 |

|  |  |  |
| --- | --- | --- |
| 7. | Chapter 07 Testing Strategy   * 1. Manual Functional testing   2. Cross-Browser Testing   3. API and Security Testing | 38-40 |
| 8. | Chapter 08 Snapshots of GUI | 41-43 |
| 9. | Chapter 09 References | 44-45 |
| 10. | Chapter 10 Brief Profile of the Student | 46-47 |

**Chapter-1**

# COMPANY PROFILE

## COMPANY PROFILE

### Introduction to NHPC Limited

NHPC Limited (erstwhile National Hydroelectric Power Corporation) is an Indian public sector hydropower company that was incorporated in 1975 to plan, promote and organise an integrated and efficient development of hydroelectric power. Recently it has expanded to include other sources of energy like solar, geothermal, tidal, and wind.

At present, NHPC is a Navaratna enterprise of the Government of India and among the top ten companies in the country in terms of investment base. Baira Suil Power station in Salooni tehsil of Chamba district was the first project undertaken by NHPC.

### Public Limited Company

NHPC is listed on the National Stock Exchange and Bombay Stock Exchange on 1 September 2009. The government of India and State Governments have 67.40% share as a promoter of the Company while remaining 32.6% is public shareholding. The total number of shareholders are 191,337 and share capital is ₹12,300,742,773.

### Market Value

At present, NHPC is a schedule 'A' Enterprise of the Govt. of India with an authorised share capital of ₹150 billion, with an investment base of over ₹552 billion Approx. In 2015–16 NHPC made a profit after tax of ₹24.40 billion. An increase of 15% than the previous year profit of ₹21.24 billion. NHPC is among the top 10 companies in India in terms of investment.

Initially, on incorporation, NHPC took over the execution of Salal Stage-I, Bairasiul and Loktak Hydro-electric Projects from Central Hydroelectric Projects Control Board. Since then, it has executed 22 hydro projects with an installed capacity of 6717 MW on ownership basis including projects taken up in a joint venture. One wind project of 50 MW has also been commissioned in October 2016. NHPC has also executed 5 projects with an installed capacity of 89.35 MW on turnkey basis. Two of these projects have been commissioned in neighbouring countries i.e. Nepal and Bhutan at a capacity of 14.1 &60

MW.

### Ongoing Projects (as of September 2023)

Presently NHPC is engaged in the construction of 3 projects aggregating to a total capacity of 3130 MW. NHPC has planned to add 1702 MW during 12th Plan period of which 1372 MW has been completed. 5 projects of 4995 MW are awaiting clearances/Govt. approval for their implementation. Detailed Projects reports are being prepared for 3 projects of 1130 MW. Besides, 3 projects of 1230 MW are under development through its JV, Chenab Valley Power Projects Pvt. Ltd. in J&K.

In late 2016, NHPC commissioned a 50 MW wind Power Project in Jaisalmer, Rajasthan.

In April 16th 2025 with commission of its 4th Unit NHPC fully commissioned 800MW Parbati-II H.E. Project.

Since its inception in 1975, NHPC has grown to become one of the largest organisations in the field of hydropower development in the

country. With its present capabilities, NHPC can undertake all activities from concept to commissioning of hydroelectric projects.

### Objectives

* To Plan, promote and organize an integrated and efficient development of power in all its aspects through Conventional and Non-Conventional Sources in India and Abroad, including planning, investigation, research, design and preparation of preliminary, feasibility and definite project reports, construction, generation, operation and maintenance of power stations and projects, transmission, distribution, trading and sale of power generated at Stations in accordance with the national economic policy and objectives laid down by the Central Government from time to time and release of water and other needs to the State Govt. as per the agreed parameters.
* To undertake, where necessary, the construction of inter-state transmission lines and ancillary works for timely and coordinated inter-state exchange of power.
* To coordinate the activities of its subsidiaries, to determine their economic and financial objectives / targets and to review, control, guide and direct their performance with a view to secure optimum utilization of all resources placed at their disposal.
* To act as an agent of Government / Public Sector financial institutions, to exercise all the rights and powers exercisable at any meeting of any Company engaged in the planning, investigation, research, design and preparation of preliminary, feasibility and definite project reports, construction, generation, operation, maintenance of Power Stations and Projects, transmission,

distribution, trading and sale of power in respect of any shares held by the Government, Public financial institutions, nationalized banks, nationalized insurance companies with a view to secure the most effective utilization of the financial investments and loans in such companies and the most efficient development of the concerned

* To carry on the business of purchasing, selling, importing, exporting, producing, trading, manufacturing or otherwise dealing in all aspects of planning, investigation, research, design and preparation of preliminary, feasibility and definite project reports, construction, generation, operation and maintenance of Power Stations and Projects, transmission, distribution and sale of Power, Power Development, including forward, backward or horizontal integration ancillary and other allied industries and for that purpose to install, operate and manage all necessary plants, establishments and works.

Authorised Capital ₹ 15000 Crore

Value of Assets ₹ 87121 Crore (As on 31.3.2025) Paid Up Capital ₹ 10045.03 Crore (As on 31.3.2025)

### Profile

* + 1. **Power Stations:**

Total: 28Nos. 7232.90 MW

Hydro (Incl- 2 in JV): 22 Nos. 6971.20 MW

Wind: 01 No. 50 MW

Solar (Incl- 4 in JV): 05 No. 211.70 MW

### Projects Under Construction:

Total: 16 Nos. 10804.00 MW

Hydro (Incl. JV): 9 Nos. 9314.00 MW

Solar: 07 Nos. 1490.00 MW

### Projects Awaiting Clearances:

Total 09 Nos. 4291 MW

Hydro (Incl.1 in JV) 5 Nos. 4046 MW

Solar (Incl. 3 in JV) 4 Nos. 245 MW

### Projects Under Survey and Investigation Stage:

Total : 10 Nos. 9715 MW

Hydro : 6 Nos. 5475 MW

Pump Storage : 04 No.

### Projects Under New Initiative:

4240 MW

|  |  |  |
| --- | --- | --- |
| Total : | 16 Nos. | 27214 MW |
| Hydro : | 3 Nos. | 13900 MW |
| Pump Storage : | 13 No. | 12690 MW |

## AWARDS AND HONOURS

NHPC Limited, India’s premier hydropower company was awarded “CBIP Award for Best Performing Utility in Hydropower Sector” at CBIP Awards on CBIP Day on 04 January 2019 in New Delhi. The award was given for outstanding contribution to the nation for development and efficient operation of Hydropower Plants in the country besides contribution for development of Hydropower in the neighbouring countries. NHPC also has been officially granted Navaratna status by the Indian government on August 30, 2024. This recognition provides NHPC with greater operational and financial autonomy, allowing it to make investments up to ₹1,000 crore without requiring government approval.

## IT&C DEPARTMENT, CORPORATE OFFICE

The Information Technology & Communication (IT&C) Division of the Corporate Office, NHPC Ltd., Faridabad, plays a pivotal role in driving

digital transformation and technological advancement across the organization. It is responsible for the planning, implementation, and management of enterprise-wide IT infrastructure, software solutions, cybersecurity, and communication systems. By ensuring robust and secure IT services, the division supports efficient decision-making, enhances operational productivity, and enables seamless collaboration across NHPC’s various projects and regional offices. Its initiatives align with NHPC’s vision of leveraging technology to strengthen its core business and administrative functions.

**Chapter-2**

# INTRODUCTION TO PROJECT

## INTRODUCTION TO PROJECT

### Overview

The **NHPC Contract Submission Portal** is a secure, internal web application engineered to modernize and centralize the management of contracts across various departments and groups within NHPC.

This portal empowers authorized employees to perform critical tasks, including submitting new contract details, uploading essential documents (such as agreements and bank guarantees), and processing contract extensions. The system is built on a robust role-based access control model, segregating data by employee groups to ensure users can only view and manage contracts relevant to their designated functions. This creates a secure, efficient, and transparent digital environment for contract lifecycle management.

### Working:

* + - **User Authentication:**

An employee initiates a session by entering their Employee ID and password on the login page. The backend uses the bcrypt library to securely hash the password and verify it against the stored hash in the database.

* + - **Session Management:**

Upon successful authentication, a secure session is established using express-session. This session stores the employee's ID, name, and department, maintaining their logged-in state across the application.

* + - **Group-Based Dashboard:**

The user is directed to a central dashboard where they select a workgroup from a dropdown menu. This selection is stored in their session and dictates the data they can access.

* + - **Data Rendering:**

The dashboard dynamically fetches and displays a list of all contracts associated with the selected group, providing an at-a-glance overview of contract IDs, statuses, and key dates.

* + - **Contract & Extension Submission:**

Employees can navigate to dedicated forms to submit new contracts or file for extensions. The backend, powered by multer, handles the secure upload of required PDF documents.

* + - **Backend Processing:**

The **Node.js** and **Express.js** backend serves as the application's core. It manages all incoming API requests, validates data, processes file uploads, and executes CRUD (Create, Read, Update, Delete) operations on the MySQL database.

### 2.3. Purpose and Objectives

The primary objective of this project is to transition from manual, paper- based contract management to a streamlined, secure, and efficient digital platform.

**Centralization:** To provide a single source of truth for all contract-related information, eliminating data silos.

**Security:** To enforce data integrity and confidentiality through encrypted passwords, secure sessions, and strict group-based access controls.

**Efficiency:** To automate the submission and tracking of contracts and extensions, significantly reducing administrative overhead and paperwork.

**Transparency & Accountability:** To offer a clear dashboard for monitoring contract lifecycles and to maintain a precise log of all actions performed by users.

**Chapter-3**

# REQUIREMENT ANALYSIS

## REQUIREMENT ANALYSIS

### Functional Requirements

1. User & Group Management:
   * **Authentication**: The system must provide a secure login mechanism for registered employees.
   * **Group selection:** Post-login, users must be prompted to select a workgroup, which dictates their access scope.
2. Contract Lifecycle Management:
   * An authorized user must be able to create a new contract, providing all necessary details (e.g., Contract ID, ERP No., Engineer-in-Charge, key dates).
   * The system must allow the upload of mandatory documents (Agreement, Bank Guarantee) in PDF format.
   * The contract status ('Active' or 'Closed') must be automatically calculated and assigned by the system based on its closure date relative to the current date.
3. Extension Management:
   * The system must allow users to submit extensions for existing, valid contracts.
   * For each extension, a new, unique ERP number must be programmatically generated (e.g., BASE\_ERP- EXT1, BASE\_ERP-EXT2).
4. Dashboard & Reporting:
   * The main dashboard must provide a summarized, tabular view of all contracts within the selected group.
   * The view must include key fields like Contract ID, ERP No., status, and relevant dates.

### 3.2Non-Functional Requirements

1. Security**:**
   * The system must enforce strong password security by hashing all user passwords using the bcrypt algorithm.
   * File uploads must be strictly validated on the server side to accept only PDF files, mitigating the risk of malicious file uploads.
   * Sensitive configuration data, including the session secret and database credentials, must be managed via environment variables (.env) and not hard-coded.
2. Performance**:**
   * The application's API response time for data retrieval should be optimized to ensure the dashboard loads quickly, even with a large number of contracts.
   * Database queries must be indexed appropriately to ensure efficient data fetching.
3. Usability**:**
   * The user interface must be intuitive, responsive, and accessible across all modern web browsers (Chrome, Firefox, Edge).
   * The system must provide clear, non-disruptive feedback for user actions (e.g., success/error modals using SweetAlert2).
4. Reliability**:**
   * The backend server must include robust error handling to prevent crashes from unexpected user input or database errors, ensuring high availability.

### 3.3 System Requirements

1. **Frontend Stack:**
   * **HTML5 & CSS3:** For structuring and styling the web pages.
   * **JavaScript (ES6):** For client-side logic, DOM manipulation, and API communication (fetch).
   * **Libraries:** SweetAlert2 (for user-friendly alerts), Flatpickr (for a modern date picker), AOS.js and Particles.js (for UI animations and aesthetics).

### Backend Stack:

* + **Node.js:** A JavaScript runtime environment chosen for its event-driven, non-blocking I/O model, ideal for building fast and scalable network applications.
  + **Express.js:** A minimalist and flexible Node.js web application framework, providing a robust set of features for web and mobile applications.
  + **Core Libraries:** mysql2 (high-performance MySQL driver), bcrypt (password hashing), express- session (session management), multer (file upload handling), dotenv (environment variable management).

### Database:

* + **MySQL Server:** A reliable, open-source relational database management system (RDBMS) chosen for its stability and widespread support.

**Chapter-4**

# DESIGN

## DESIGN

The system is designed with a clear separation of concerns, following modern web development principles to ensure security, scalability, and maintainability. The design is broken down into a high-level overview of the components and a detailed data flow for each core function.

### High-Level Design

* + 1. **Frontend:**
       - **Technology:** The frontend is built using standard **HTML5, CSS3,** and **JavaScript (ES6)**, ensuring broad browser compatibility.
       - **User Experience (UX):** It leverages specialized JavaScript libraries to create a rich user experience:
         * **SweetAlert2** is used for modern, non-blocking alerts and confirmation dialogs.
         * **Flatpickr** provides a user-friendly and highly customizable date/time picker.
         * **AOS.js** and **Particles.js** are used for aesthetic animations and background effects.
       - **Communication:** Communicates exclusively with the backend via asynchronous **AJAX (fetch API)** requests, allowing for dynamic content updates without requiring page reloads.

### Backend:

* + - * **Technology:** The server is built on the **Node.js** runtime using the **Express.js** web framework.
      * **API Endpoints:** It exposes a RESTful API to the frontend, with key endpoints including:
        + /login for user authentication.
        + /api/user and /api/select-group for user and group management.
        + /contracts/submit for creating new contracts.
        + /extensions/submit for adding contract extensions.
        + /contracts/by-group for fetching contract data.

### Core Functionality:

* + - * + Uses the **multer** middleware for robustly handling multipart/form- data uploads, specifically for processing PDF files.
        + Uses the **bcrypt** library for securely hashing and comparing user passwords.
        + Returns **JSON** payloads for data requests and status messages.

### General System Data Flow:

* + - * The user authenticates through the frontend.
      * The frontend communicates the user's selected workgroup to the backend, which is stored in a session.
      * The user submits forms (with or without files) via the frontend.
      * The backend validates the request, processes files, interacts with the MySQL database for persistence, and returns a status response.
      * The frontend receives the response and dynamically updates the UI to reflect the result of the action.

### Detailed Data Flow Design

This section breaks down the step-by-step process for each primary feature of the application.

### User Authentication and Session Flow

1. **User Action:**
   * The user enters their Employee ID and password into the login.html form.
   * The user clicks the "Login" button.

### Frontend Processing:

* + An onsubmit event listener prevents the browser's default form submission.
  + JavaScript collects the input values and constructs a JSON object (e.g., { "emp\_id": "...", "password": "..." }).

### API Request:

* + The frontend sends an asynchronous POST request to the /login endpoint with the JSON object in the request body.

### Backend Processing:

* + The Express server receives the request.
  + It retrieves the user's record from the employees table in the MySQL database based on the emp\_id.
  + It uses bcrypt.compare() to securely check if the provided password matches the stored hash.
  + If the credentials are valid, it initializes a new session using express- session and stores the user's details (e.g., emp\_id, emp\_name) in the req.session object.

### API Response:

* + **On Success:** The server sends a 200 OK status with a JSON response (e.g., { "message": "Login successful" }).
  + **On Failure:** The server sends a 401 Unauthorized status with a relevant error message (e.g., { "message": "Wrong password" }).

### Frontend Response:

* + JavaScript handles the API response.
  + It uses SweetAlert2 to display a "Success" or "Error" modal to the user based on the response.
  + Upon a successful login, the user is redirected to dashboard.html.

### Diagram:

[User] --(Enter Credentials)--> [Frontend] --(POST /login, JSON)--> [Express API]

<--(200 OK / 401 Unauthorized, JSON)-- [Express API]

### New Contract Submission Flow

1. **User Action:**
   * The user fills out all the required fields in the contract-form.html.
   * The user selects the required PDF files (Agreement, Bank Guarantee) using the <input type="file"> elements.
   * The user clicks the "Submit Contract" button.

### Frontend Processing:

* + JavaScript captures all form data, including the selected files, and bundles it into a FormData object. This object is necessary for sending multipart/form-data.

### API Request:

* + The frontend sends a POST request to the /contracts/submit endpoint with the FormData object as the payload.

### Backend Processing:

* + The multer middleware intercepts the request. It parses the multipart/form-data, saves the uploaded PDF files to the /uploads/ directory on the server with unique names, and attaches the file information to the req object.
  + The request proceeds to the main controller function.
  + The controller extracts the text-based form fields from req.body and the generated file paths from req.files.
  + It constructs and executes a SQL INSERT statement to save the complete contract record into the contracts table in the database.

### API Response:

* + The server sends back a JSON response with a success message (e.g., { "message": "Contract submitted successfully" }).

### Frontend Response:

* + JavaScript displays a SweetAlert2 success modal to inform the user that the contract was submitted.
  + The user is typically redirected back to the dashboard to see the new entry.

### Diagram:

[User] --(Fill Form & Upload Files)--> [Frontend] --(POST /contracts/submit, FormData)--> [Express API (Multer -> Controller)]

<--(200 OK, JSON)-- [Express API]

### Dashboard Data Loading Flow

1. **User Action:**
   * After logging in, the user views the dashboard.html page.
   * The user selects a workgroup from the #groupSelector dropdown menu.

### Frontend Processing:

* + A change event listener on the dropdown is triggered.

### API Request:

* + The frontend makes two sequential API calls:
    1. A POST request to /api/select-group with the chosen group\_id to update the user's session on the backend.
    2. Upon success, a GET request to /contracts/by-group to fetch the contracts for the now-active group.

### Backend Processing:

* + The /api/select-group endpoint updates the req.session.selected\_group\_id value.
  + The /contracts/by-group endpoint reads the selected\_group\_id from the session and executes a SQL SELECT query to retrieve all matching contracts from the database.

### API Response:

* + The server sends a 200 OK response containing a JSON array of contract objects.

### Frontend Response:

* + JavaScript receives the JSON array.
  + It clears the existing content of the contract table's <tbody>.
  + It iterates through the array and dynamically creates a new table row (<tr>) for each contract object, populating the cells (<td>) with the relevant data.
  + The new rows are appended to the table, displaying the contracts to the user.

### Diagram:

[User] --(Select Group)--> [Frontend] --(POST /api/select-group)--> [Express API]

<--(200 OK)-- [Express API]

[Frontend] --(GET /contracts/by-group)--> [Express API]

<--(200 OK, JSON Array)-- [Express API]

**Chapter-5**

# MODULE IMPLEMENTATION

## MODULE IMPLEMENTATION

The implementation of the NHPC Contract Submission Portal was divided into logical modules, each responsible for a distinct set of functionalities. This modular approach facilitated development, testing, and future maintenance.

### Frontend UI and Interaction Module

This module encompasses the overall design and user experience of the application. It was built not as a single-page application, but as a collection of interconnected pages that provide a seamless, app-like experience.

* **Designed and implemented a responsive, multi-page interface** using standard HTML5, CSS3, and JavaScript, ensuring a consistent and accessible experience across different devices and screen sizes.
* **Developed a custom CSS theme** featuring a modern "glassmorphism" aesthetic, with support for both **light and dark modes** to enhance user comfort. The user's theme preference is saved in localStorage to persist across sessions.
* **Integrated SweetAlert2** to replace standard browser alerts. This library provides elegant, non-blocking modals for success messages, error notifications, and confirmation dialogs (e.g., for logging out or selecting a workgroup), significantly improving the user experience.
* **Implemented Flatpickr** for all date input fields, providing a clean, user- friendly, and consistent date selection widget across the application.
* **Enhanced the user interface with subtle animations** using AOS.js for on-scroll reveals and Particles.js for a dynamic, professional background, creating a more engaging visual experience.
* **Implemented client-side validation** using HTML5 required attributes and JavaScript checks to ensure that critical data, such as contract IDs and group selections, are provided before form submission or navigation.

### Contract Submission Module

This module corresponds to the contract-form.html page and its associated logic. It is the primary interface for entering new contract data into the system.

* **Allows users to input comprehensive contract details**, including Contract ID, ERP Number, Engineer-in-Charge, and all relevant dates (award, start, end, closure).
* **Enables the secure upload of mandatory PDF documents**, specifically the Contract Agreement and the Bank Guarantee, using standard file input fields.
* **Features a dynamic and intelligent user interface**. The form's visibility changes based on user input; for example, selecting "Yes" for the "Is Extended?" dropdown conditionally displays the "New Closure Date" field and replaces the standard "Submit" button with a "Submit & Go to Extension" button, guiding the user through a logical workflow.
* **Handles the seamless transition to the extension module**. When a user creates an extended contract, the system first submits the new contract and then automatically redirects the user to the extension form, pre- populating the new Contract ID to prevent errors.

### Contract Extension Module

This module is handled by the extension-form.html page and is specifically designed for adding extensions to existing contracts.

* **Allows users to submit a new extension** by providing the extension date, a new bank guarantee document, and other relevant information.
* **Ensures data integrity by pre-populating the Contract ID**. The page is designed to receive the contract\_id via a URL query parameter. This value is then placed into a read-only input field, which guarantees that the extension is linked to the correct parent contract.
* **Provides a focused and streamlined workflow** by presenting only the fields necessary for an extension, reducing complexity and potential for user error.

### Dashboard and Data Display Module

This module represents the core functionality of the dashboard.html page, which serves as the user's central hub after logging in.

* **Implements a mandatory group selection workflow**. Upon landing on the dashboard, the user must select a workgroup from a dropdown menu. This selection is stored in the session and acts as the primary filter for all data displayed on the page.
* **Dynamically renders contract data**. After a group is selected, the frontend makes an AJAX (fetch) call to the backend. The returned array of contract objects is then used to dynamically build and populate the main contracts table in the UI, ensuring the user always sees real-time, relevant data.
* **Provides guided navigation**. The "New Contract" and "Extend Contract" buttons use SweetAlert2 to prompt users to select a group if they belong to more than one, ensuring that every new entry is correctly categorized.

### Backend API and Logic Module

This module forms the server-side core of the application, built with Node.js and Express.js. It handles all business logic, security, and database interactions.

* **Developed Express.js endpoints** for all application functions, including /login, /contracts/submit, /extensions/submit, /api/user, and /contracts/by-group.
* **Handles secure file uploads** using the **multer** middleware. This includes a custom file filter that strictly validates and accepts only files with

a .pdf extension, enhancing application security. It also generates unique filenames to prevent collisions.

### Implements robust security measures:

* + Uses **bcrypt** to hash all user passwords before they are stored in the database and to securely compare passwords during login.
  + Uses **express-session** to create and manage secure, server-side user sessions, ensuring that user authentication is maintained and protected.
* **Executes all business logic**, such as automatically determining a contract's status based on its closure date and programmatically generating unique ERP numbers for new extensions.
* **Implements comprehensive error handling** and returns clear, meaningful JSON responses to the frontend for both successful and failed operations.

**Chapter-6**

# DATABASE AND DATA DICTIONARY

## DATABASE AND DATA DICTIONARY

A relational database model was designed using MySQL to ensure data integrity, minimize redundancy, and establish clear relationships between the application's core entities. The schema is normalized and consists of five primary tables.

### Database Schema

* + 1. **employees Table**

This table stores the essential information for each employee user, including their credentials and permissions.

**Column Name**

**Data Type Key/Const raint**

Primary

**Descrip tion**

The unique identifie

emp\_id INT

VARCHAR(2

Key

r for each employe e.

The full name of

emp\_name

department

password

contract\_ent ry

55)

VARCHAR(2 55)

VARCHAR(2 Not Null 55)

ENUM('YES',

'NO')

the employe e.

The departm ent the employe e belongs to.

Stores the securely hashed passwor d using bcrypt.

Permissi on flag to determi ne if the

approve\_co ntract

* + 1. **groups Table**

ENUM('YES',

'NO')

user can create contract s.

Permissi on flag for contract approva l capabilit ies.

This table acts as a simple lookup for the different workgroups or departments.

**Column Name**

**Data Type Key/Constra int**

**Descripti on**

The unique

group\_id INT Primary Key

identifier for each group.

The

group\_na me

VARCHAR(2 Unique 55)

unique name of the group.

* + 1. **employee\_groups Table**

This is a linking (or join) table that establishes a many-to-many relationship between employees and groups.

|  |  |  |  |
| --- | --- | --- | --- |
| **Colu mn Name** | **Da ta Ty pe** | **Key/Constr aint** | **Description** |
| emp\_i | IN | Composite Primary | References employees(e |
| d | T | Key, Foreig n Key | mp\_id). |
| group | IN | Composite Primary | References groups(group |
| \_id | T | Key, Foreig  n Key | \_id). |

* + 1. **contracts Table**

This is the central table of the application, storing all metadata for each contract.

**Column Data Type Name**

contract\_id VARCHAR (255)

erp\_no VARCHAR (255)

**Key/C onstra int**

Primar y Key

Not Null

**Description**

The unique business identifier for the contract.

The ERP number associated with the contract.

Name of the

engineer\_i ncharge

contract\_a

VARCHAR (255)

engineer responsible for the contract.

The date the

ward\_date DATE

start\_date DATE

end\_date DATE

contract was awarded.

The official start date of the contract.

The official end date of the contract.

agreement\_ file

agreement\_ submit\_dat e

agreement\_ accept\_dat e

VARCHAR (255)

DATE

DATE

The filename of the uploaded agreement PDF.

The date the agreement was submitted.

The date the agreement was accepted.

The filename of

bg\_file VARCHAR (255)

the uploaded bank guarantee PDF.

bg\_submit DATE

\_date

bg\_accept\_ DATE date

extended ENUM('YE S','NO')

closure\_dat DATE e

new\_closur DATE e\_date

ENUM('Act

Defaul t: 'NO'

Defaul t:

The date the bank guarantee was submitted.

The date the bank guarantee was accepted.

Flag indicating if the contract has been extended.

The original date of contract closure.

The new closure date after an extension.

The automatically

status

ive','Closed'

)

'Close d'

Foreig

determined

status of the contract.

References grou ps(group\_id).

group\_id INT

emp\_id INT

* + 1. **extensions Table**

n Key

Foreig n Key

Links the contract to a group.

References empl oyees(emp\_id). Tracks which employee submitted it.

This table stores records for each extension applied to a contract, representing a one-to-many relationship with the contracts table.

**Column Name**

**Data Type**

**Key/C onstrai nt**

**Description**

The unique

id INT Primar y Key

identifier for the extension record itself.

contract\_id

VARC HAR(2 55)

Foreig n Key

References contrac ts(contract\_id).

Links the extension to its parent contract.

The sequential

extension\_no INT Not Null

extension\_da DATE te

VARC

number of the extension (e.g., 1,

2, 3).

The date the extension was granted.

The filename of

bg\_extended

\_file

bg\_extended

HAR(2 55)

the new bank guarantee PDF for the extension.

The submission

\_date DATE

bg\_extended DATE

\_accept\_date

VARC

date of the new bank guarantee.

The acceptance date of the new bank guarantee.

The unique ERP

erp\_no

HAR(2 55)

number generated for this specific extension.

group\_id INT Foreig n Key

References groups (group\_id).

### Entity-Relationship Description

* **Employees and Groups (Many-to-Many):** An employee can be a member of multiple groups, and a group can have multiple employees. This relationship is implemented via the employee\_groups linking table. This is the foundation of the system's group-based access control.
* **Contracts and Extensions (One-to-Many):** A single contract can have multiple extensions over its lifetime. This is modeled by

the extensions table having a foreign key (contract\_id) that points to the contracts table.

* **Groups and Contracts (One-to-Many):** A single group can be responsible for many contracts, but each contract belongs to only one group. This is defined by the group\_id foreign key in the contracts table.

**Chapter-7**

# TESTING STRATEGY

## TESTING STRATEGY

A multi-layered testing strategy was employed to ensure the application's functionality, security, and reliability. This included manual testing of user workflows, cross-browser checks, and direct API verification.

### Manual Functional Testing

This phase focused on validating the end-to-end user experience for all core features.

* **Authentication:** Verified that users could log in with valid credentials and were blocked with specific error messages for invalid attempts.
* **Form Submissions:** Tested the "New Contract" and "Extension" forms with both valid and incomplete data. Confirmed that submissions were successful with correct data and that required field validations worked as expected.
* **Dashboard Functionality:** Ensured that after a user selected a group, the contract table correctly populated with data only from that group. Confirmed that switching between groups correctly refreshed the data table.
* **File Uploads:** Validated that only files with a .pdf extension could be selected for upload in the forms.

### Cross-Browser and Accessibility Testing

* **Cross-Browser:** Verified that the application's layout and core functionalities (login, form submissions, dashboard rendering) worked consistently across the latest versions of **Google Chrome, Mozilla Firefox,** and **Microsoft Edge**.
* **Accessibility:** Performed keyboard navigation checks on all forms. Confirmed that all interactive elements were focusable and could be operated using the Tab and Enter keys.

### API and Security Testing

This phase focused on the backend's robustness and security, primarily using Postman for direct API requests.

### API Logic:

* + Sent requests with incorrect or missing data to API endpoints to verify server-side validation and error handling.
  + Checked that endpoints returned the correct HTTP status codes (200 OK, 401 Unauthorized, 400 Bad Request) based on the request's validity.

### Authorization:

* + Attempted to access protected routes (e.g., /contracts/by-group) without a valid session cookie and confirmed that access was denied with a 401 Unauthorized error.
  + Verified that a user belonging only to "Group A" could not fetch data for "Group B" by manually crafting an API request.

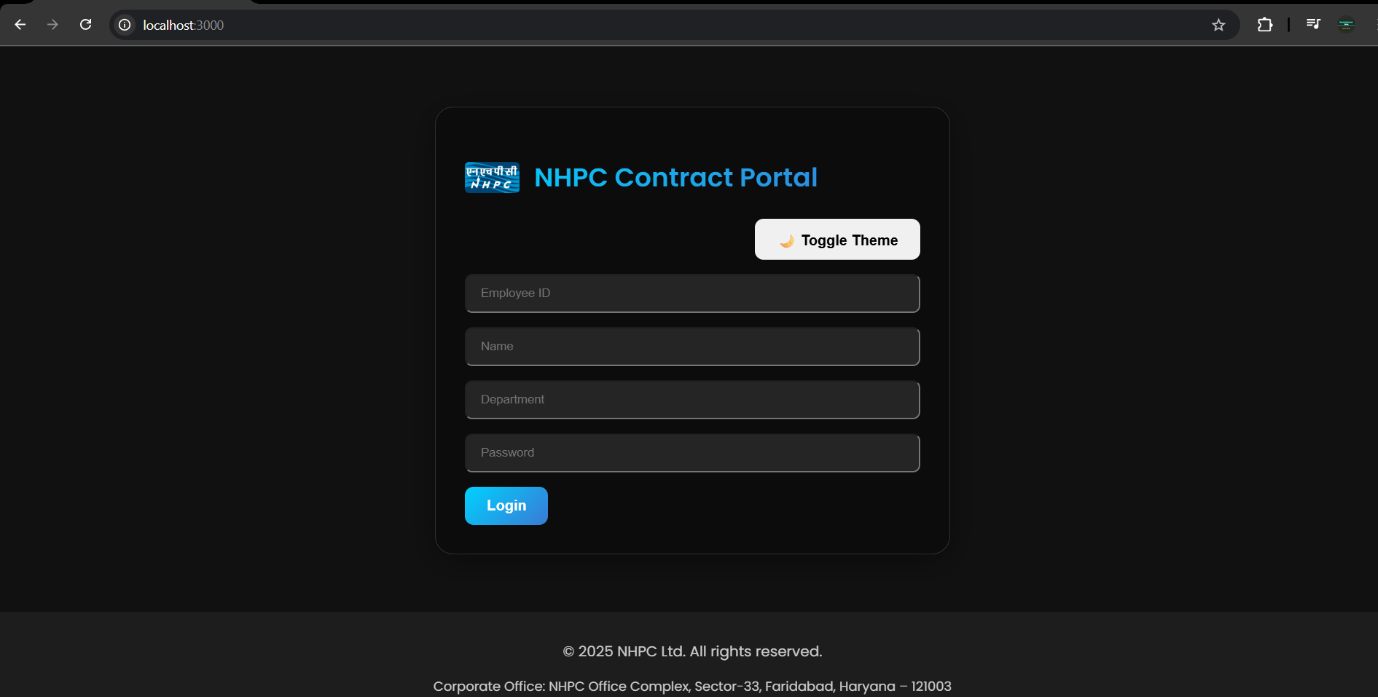
### Security Verification:

* + Inspected the employees table in the database to confirm that all passwords were correctly hashed by **bcrypt** and not stored in plaintext.
  + Confirmed that the multer file upload middleware correctly rejected any files that were not in PDF format.

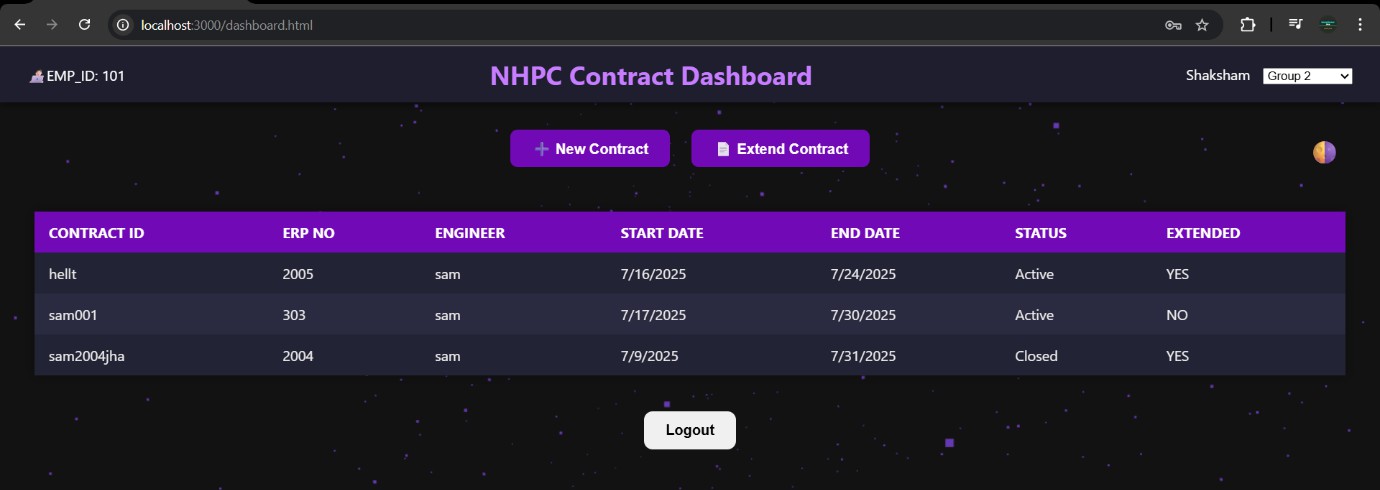
**Chapter-8**

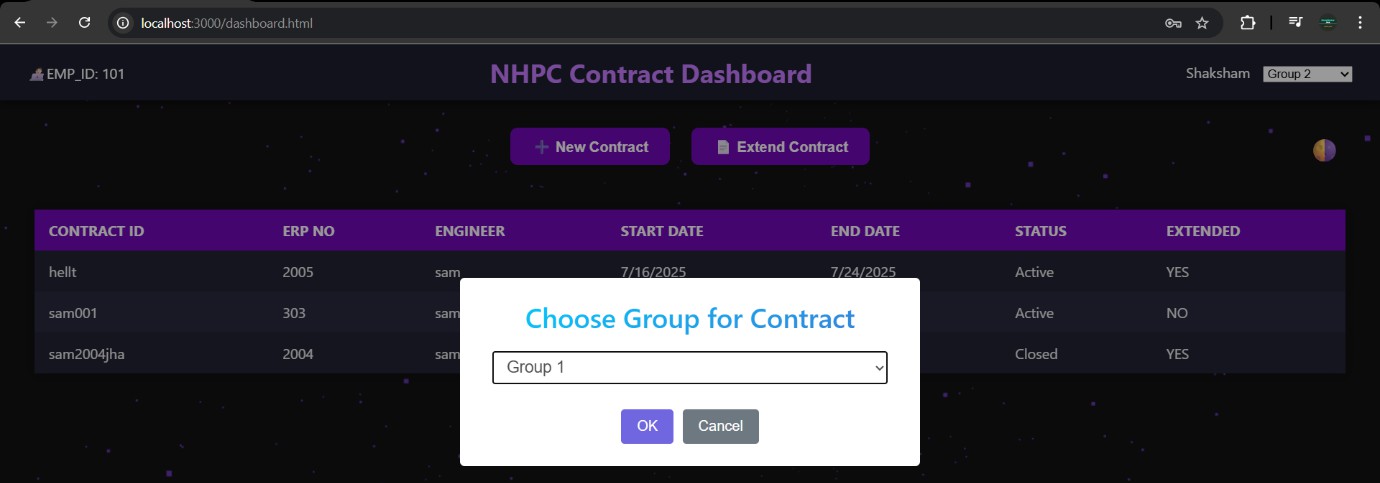
# SNAPSHOTS OF GUI

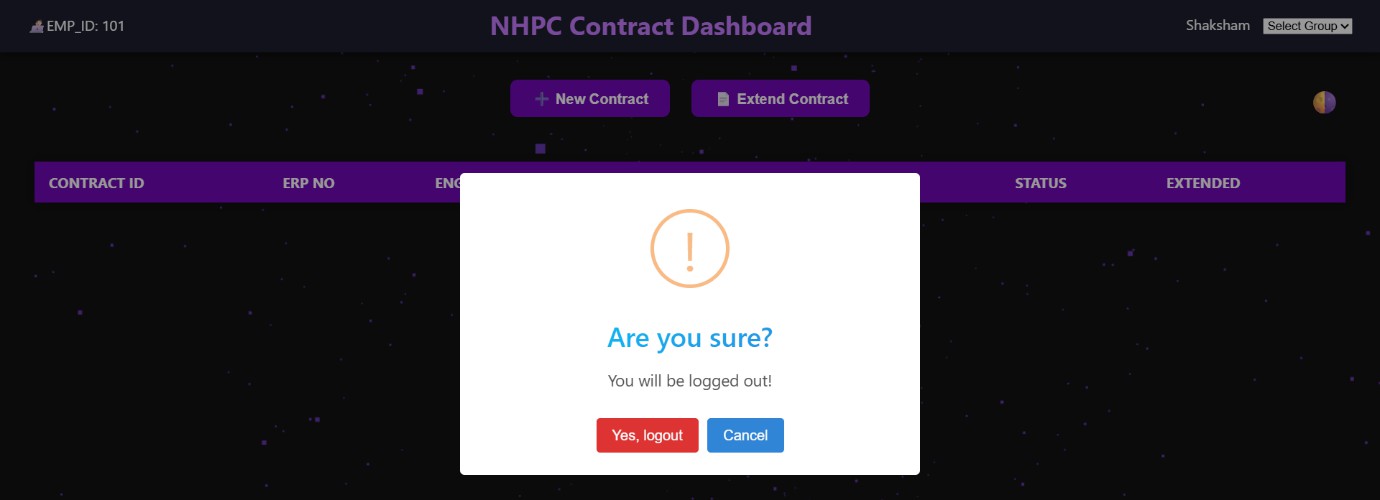
1. **SNAPSHOTS OF GUI**
   1. **USER LOGIN PAGE**

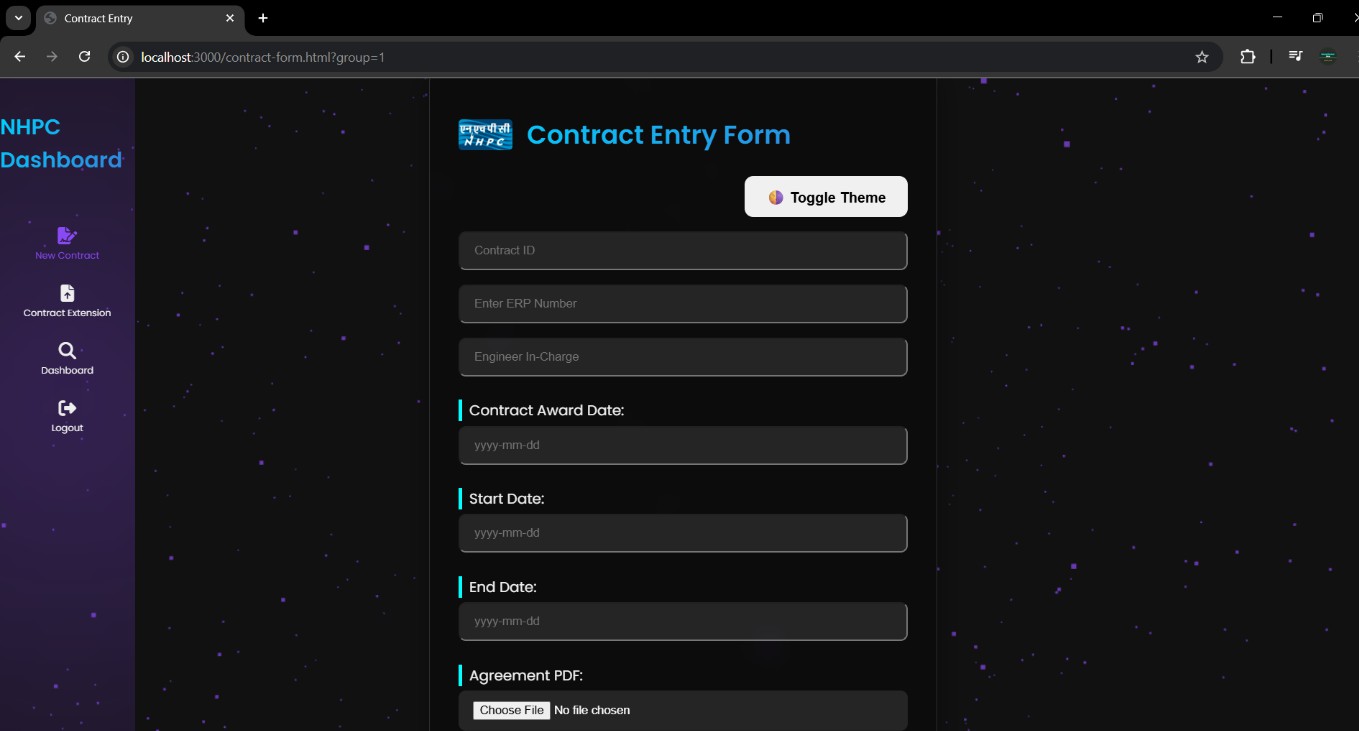
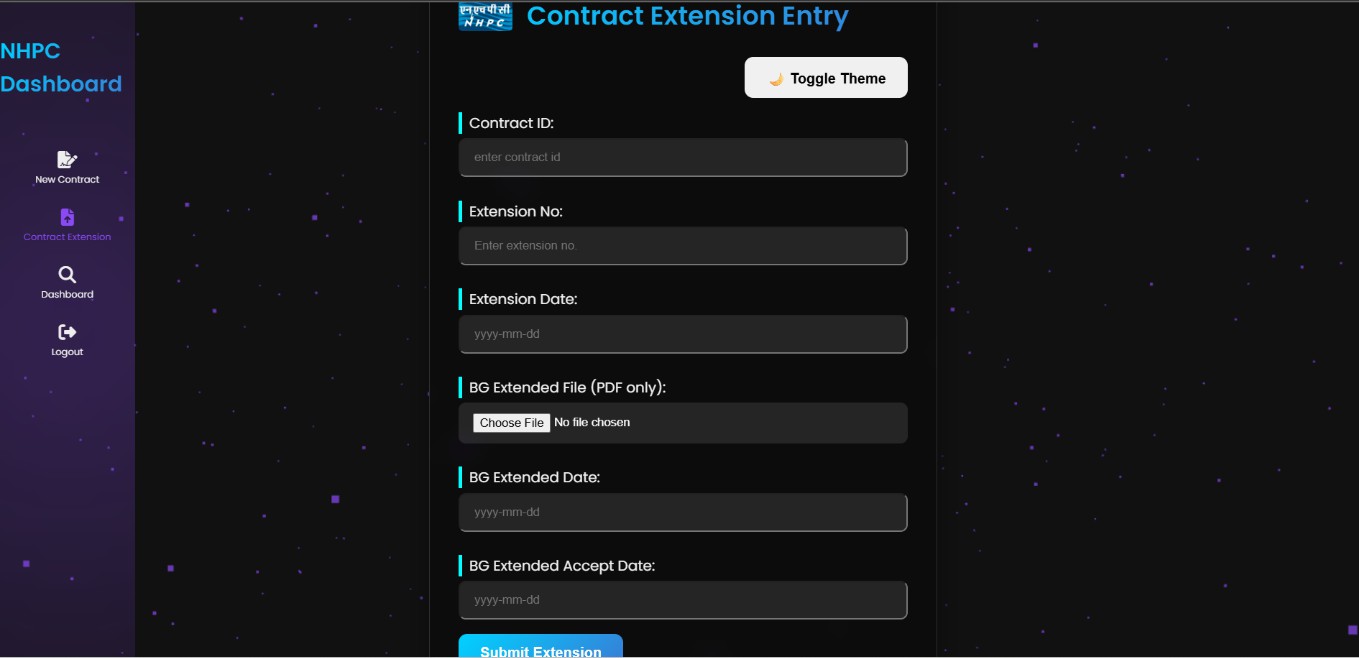
****

* 1. **DASHBOARD**



****



* 1. **CONTRACT SUBMISSION PAGE**
  2. **CONTRACT EXTENSION PAGE**

**Chapter-9**

# REFERENCES

## REFERENCES

This section provides a list of the key technologies, libraries, and frameworks that were instrumental in the development of the NHPC Contract Submission Portal, along with links to their official documentation.

**Core Backend Technologies Node.js:** [https://nodejs.org/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fnodejs.org%2F)

Official documentation for the Node.js runtime environment, which provided the foundation for the application's server-side logic.

**Express.js:** [https://expressjs.com/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fexpressjs.com%2F)

Official documentation for Express.js, the minimalist web framework used to build the backend server, manage routing, and handle middleware.

**MySQL:** [https://www.mysql.com/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fwww.mysql.com%2F)

Official documentation for the MySQL Database Server, which was used for persistent data storage.

**Key NPM Packages**

**Nodemon:** [https://www.npmjs.com/package/nodemon](https://www.google.com/url?sa=E&q=https%3A%2F%2Fwww.npmjs.com%2Fpackage%2Fnodemon)

Documentation for Nodemon, a critical development utility used to automatically restart the server during development, improving workflow efficiency.

**Bcrypt.js:** [https://www.npmjs.com/package/bcrypt](https://www.google.com/url?sa=E&q=https%3A%2F%2Fwww.npmjs.com%2Fpackage%2Fbcrypt)

Official documentation for the bcrypt library, which was used for securely hashing and comparing user passwords, a cornerstone of the application's security model.

**Multer.js:** [https://github.com/expressjs/multer](https://www.google.com/url?sa=E&q=https%3A%2F%2Fgithub.com%2Fexpressjs%2Fmulter)

Documentation for Multer, the middleware used to handle multipart/form-data, which was essential for processing PDF file uploads.

**Express-session:** [https://www.npmjs.com/package/express-session](https://www.google.com/url?sa=E&q=https%3A%2F%2Fwww.npmjs.com%2Fpackage%2Fexpress-session) Documentation for the express-session library, used to create and manage secure, server-side user sessions for authentication.

**Frontend Libraries**

**SweetAlert2:** [https://sweetalert2.github.io/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fsweetalert2.github.io%2F)

Official documentation for SweetAlert2, the library used to create responsive, accessible, and highly customizable alerts and modals.

**Flatpickr:** [https://flatpickr.js.org/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fflatpickr.js.org%2F)

Documentation for Flatpickr, a lightweight and powerful date/time picker used to enhance the user experience in all date-related form fields.

**Chapter-10**

# BRIEF PROFILE OF STUDENT

## BRIEF INTRODUCTION OF THE STUDENT 10.1. STUDENT DETAILS

Student Name:- SHAKSHAM Roll No:- 02720803123

Branch:- Infromation Technology

Email :- [hellt5409@gamil.com](mailto:hellt5409@gamil.com)

Phone no:- 9868572150

Address:- Plot no.7-D, Gali no.11/13,Mahadev Colony, B-Block Gurunanak Dev Colony, Bhalswa, Delhi-110042

## PROJECT DETAILS

Project Title :- NHPC CONTRACT SUBMISSION PORTAL

Project Duration :- 11th june 2025 – 16th July 2025

## ORGANIZATION DETAILS

Organization Name:- NHPC LIMITED

Address:- Santosh Nagar, Sector 33, Faridabad, Haryana 121003

Website:- <https://www.nhpcindia.com/>

## MENTOR DETAILS

Name :- Mr. TAPAN BANERJEE

Phone no:- 9891082001

Email :- [tapanBANERJEE@nhpc.co.in](mailto:tapanBANERJEE@nhpc.co.in)